REMARKS

Claims 1-36 are pending in the above-identified application. Claims 1-36 were rejected. With this Amendment, claims 1-3, 5-15, 17-27 and 29-36 were amended and claims 4, 16, and 28 were canceled without prejudice. Accordingly, claims 1-3, 5-15, 17-27 and 29-36 remain at issue.

Initially, Applicants respectfully request that the Attorney Docket No. be changed from 0007056-0233/P6791 to 30014200-1069.

I. Double Patenting Rejection Of Claims

The Examiner provisionally rejected claims 3, 15 and 27 under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claims 1, 10 and 19 of co-pending Application No. 10/021,915, entitled "Application Programming Interface for Connecting a Platform Independent Plug-In to a Web Browser."

Applicants submit that neither this Application nor co-pending Application No. 10/021,915 have allowed claims to date. Therefore, Applicants assert that it is premature to file a terminal disclaimer to overcome the provisional rejection. If the situation arises that there are claims allowed claims in this Application or Application No. 10/021,915, Applicants will revisit the issue of whether a terminal disclaimer is necessary. Accordingly, Applicants respectfully decline the Examiner's invitation to file a terminal disclaimer at this time.

II. 35 U.S.C. § 103 Obviousness Rejection of Claims

Claims 1-5, 9-17, 21-29, and 33-36 were rejected under 35 U.S.C. § 103(a) as being purportedly unpatentable over *Narin et al.*, U.S. Patent No. 6,691,176 and further in view of Admitted Prior Art (APA). In addition, Claims 6-8, 18-20 and 30-32 were rejected under 35

U.S.C. § 103(a) as being purportedly unpatentable over *Narin* in view of Admitted Prior Art (APA) and in view of NS (Netscape Gecko Technologies Enabling the Next-Generation Internet). Applicants respectfully traverse these rejections.

In the Office Action, the Examiner cites "Admitted Prior Art" against claims 1, 6-8, 10, 18-20, 22, 30-32 and 34. Applicants respectfully traverse the Examiner's assertion of "Admitted Prior Art." Applicants never admitted that the cited portions of the specification were "prior art" as defined by law (See M.P.E.P. 2129 and 35 U.S.C. § 102).

With respect to independent Claim 1 as amended and with reference to exemplary Figures 2 and 3, Applicants teach, for example, an Application Programming Interface (API) that includes a non-scriptable plug-in API (200), a scriptable plug-in API (240), and a plurality of bridges (280-281) operatively configured to connect said scriptable and said non-scriptable plug-in APIs such that a scriptable plug-in program is able to access the non-scriptable plug-in API in response to implementing the scriptable plug-in API. (See Application, pg. 9, line 13 - pg. 10 line 2; pg. 14; Figs. 2-3). Independent claims 13, and 25 have limitations similar to claim 1.

In particular, Applicants teach that in one embodiment, for example, the bridges may be a set of interfaces and a set of C++ wrappers used to bind the scriptable plug-in API to the non-scriptable plug-in API so that a plug-in program for a Web Browser like the Mozilla/Netscape 6.x browser is able to access pre-existing a non-scriptable plug-in API, such as a non-scriptable plug-in API written for Microsoft Internet Explorer. (See Application at page 10, lines 10-15).

This is clearly unlike *Narin*, which addresses a different problem than the present Application and teaches away from creating a bridge to connect a non-scriptable plug-in API and a scriptable plug-in API so that, for example, a Browser plug-in may implement and reuse the functionality of the otherwise incompatible non-scriptable plug-in API. *Narin* addresses the problem of lack of state persistence for a Web page object, which is typically created (using ActiveX Controls for Microsoft Internet Explorer Browser or scriptable Plug-Ins for the Netscape Browser) when the client navigates to the Web page that uses the object and then is destroyed when the client navigates to another Web page. (See *Narin*, Col. 6:35-61). *Narin* solves "this problem by providing a service manager for managing the life-time of services [e.g., object created from Microsoft ActiveX Controls or Netscape Plug-In depending on the Browser brand] by providing a connector object, which is the interface between the [respective type] browser scripting space and the service manager." (*Narin*, Col. 6:45-49). But *Narin* discloses that "it is difficult and time consuming to create an object that will work in different browser brands because the interface to the scripting space is different from brand to brand." (*Narin*, Col. 6:50-53). So *Narin* teaches "providing two forms for a connector object: an Active X control to be used with Microsoft's [Internet Explorer], and a Plug-In to be used with Netscape's Navigator." (*Narin*, Col. 6:59-61). Thus, *Narin* teaches away from creating an application programming interface (API) that has a bridge for operatively connecting a non-scriptable plug-in API (e.g., an ActiveX Control for Internet Explorer) to a scriptable plug-in API (e.g., Javascript Plug-in for Netscape) such that a scriptable plug-in program (e.g., software component designed for Netscape) is able to access and use the non-scriptable plug-in API (e.g., software component designed for Internet Explorer) in response to implementing the scriptable plug-in API as taught and claimed by the Applicants.

Accordingly, Applicants submit that *Narin,* alone or in combination with other cited references, fails to teach all the limitations of claims 1, 13, and 25, and respectfully request that the rejection to these claims be withdrawn.

Claims 2, 14, and 26 depend from independent claims 1, 13, and 25, respectively. Thus, claims 2, 14, and 26 should be deemed allowable for at least the same reasons as given for claims 1, 13, and 25.

With respect to independent Claim 3 as amended, Applicants teach an Application Programming Interface (API) that includes, among other limitations, "a cross platform language API" that is used to operatively bridge or connect a non-scriptable plug-in API and "a scriptable language API" that are otherwise incompatible via respective interfaces. Independent claims 15 and 27 have limitations similar to claim 3. For example, Applicants further teach that in one embodiment the cross platform language API may be an XPCOM application that "allows software components written in various programming languages to communicate with one another within the browser environment." (See Application, at pg. 10 line 21 - pg. 11 line 3).

The Examiner argues that the service manager 190 disclosed in *Narin* is a cross platform language API as claimed by Applicants. Applicants respectfully disagree. *Narin* teaches using a "connector object (either an ActiveX control interface or Plug-In interface, <u>depending on browser brand</u>) [to] translate [for] the service manager..." (See *Narin*, Col. 5:64-67, emphasis added). Thus, the *Narin* service manager 190 requires a connector object for each browser brand or programming language and, thus, does not provide "a cross platform language API" as claimed by the Applicants.

Accordingly, Applicants submit that *Narin,* alone or in combination with other cited references, fails to teach all the limitations of claims 3, 15, and 27, and respectfully request that the rejection to these claims be withdrawn.

Claims 5-10, 17-22, and 29-34 depend from independent claims 3, 15, and 27, respectively. Thus, claims 5-10, 17-22, and 29-34 should be deemed allowable for at least the same reasons as given for claims 3, 15, and 27.

With respect to independent Claim 11 as amended and with reference to exemplary Figures 6A and 6B, Applicants teach, for example, a scriptable plug-in API that includes a "scriptable plug-in" 600 and a proxy support interface (e.g., non-scriptable interface support proxy or "nsISupports Proxy" in Fig. 6B) that is operatively configured to allow the scriptable plug-in 600 to perform inter-thread calls through said proxy support, such as an inter-thread call to a non-scriptable plug-in. Independent claims 23 and 35 have limitations similar to claim 11.

Narin discloses that "the service manager [190] and interface [(either ActiveX control interface 195 or Plug-In interface 196)] act as a proxy ....on behalf of the scripting space" of the respective browser. (See *Narin*, Col. 14:43-46; Fig. 8B). *Narin* further discloses that the service manager and interface proxy "are provided to form a communication line [i.e., a single thread] between the scripting space and the services" of the ActiveX control or Plug-In. (See *Narin*, Col. 14:41-43; Fig. 8B). But *Narin* fails to teach or suggest that the service manager and interface proxy allows a scriptable plug-in to perform an inter-thread call, for example, to another scriptable plug-in or to a non-scriptable plug-in as taught and claimed by the Applicants.

Accordingly, Applicants submit that *Narin,* alone or in combination with other cited references, fails to teach all the limitations of claims 11, 23, and 35, and respectfully request that the rejection to these claims be withdrawn.

Claims 12, 24, and 36 depend from independent claims 11, 23, and 35. Thus, each of these dependent claims should be deemed allowable for at least the same reasons as given for claims 11, 23, and 35.

III. Conclusion

In view of the above amendments and remarks, Applicant submits that claims 1-3, 5-15, 17-26 and 29-36 are clearly allowable over the cited prior art, and respectfully requests early and favorable notification to that effect.

Respectfully submitted,

Dated: February 18, 2005

By: Thomas J. Burton

Thomas J. Burton
Registration No. 47,464
SONNENSCHEIN NATH & ROSENTHAL LLP
P.O. Box 061080
Wacker Drive Station, Sears Tower
Chicago, Illinois 60606-1080
(312) 876-8000